FOR ARGUMENT

Supreme Court, U.S.
FILED
APR 7 1978
MICHAEL RODAK, JR., CLERK

IN THE

Supreme Court of the United States

October Term, 1977.

No. 77-642.

LUTRELLE F. PARKER, Acting Commissioner of Patents and Trademarks,

Petitioner,

v.

DALE R. FLOOK,

Respondent.

On Writ of Certiorari to the United States Court of Customs and Patent Appeals.

BRIEF AMICUS CURIAE FOR APPLIED DATA RESEARCH, INC. (ADR).

CAROL A. COHEN,
Attorney for Amicus Curiae Applied Data Research, Inc.

International Printing Co., 711 So. 20th St., Phila., Pa. 19146 — Tel. (215) 735-8711

TABLE OF CONTENTS.

	Page
QUESTION PRESENTED	1
INTEREST OF AMICUS CURIAE	2
SUMMARY OF ARGUMENT	5
ARGUMENT	6
1. The economics of the computer industry, as well as the opposing economic interests of the competing companies in the computer industry, demonstrate that the real issues involved in this case are machines and machine systems, and not "mathematical algorithms" .	6
2. The facts of the technology and the findings of fact of the C. C. P. A. in its predecessor cases are that a programmed digital computer is a special-purpose computer or machine that performs machine functions, not a "mathematical algorithm"	9
Examples of Machine Systems Implemented by Software:	
A. AUTOFLOW Flowcharting System	11
B. ADR's Sorter Patent	14
C. The STAR System	15
D. Recognition of Printed Characters	16
E. Voice Assembly	17
3. AUTOFLOW is a machine system that could be converted into a hardwired hand-held machine but for economic reasons	19
4. The proposals of IBM and the other hardware manufacturers for copyright or other forms of protection of software would reverse the constitutional policy of an incentive to innovation in the "useful Arts"	21
CONCLUSION	23

TABLE OF CITATIONS.

Cases: Page

Dann v. Johnston, 425 U. S. 219 ..... 20, 22
Graham v. John Deere Co., 383 U. S. 1 (1966) 21
In re Bernhart, 417 F. 2d 1395 (1969) 9, 18
In re Foster, 438 F. 2d 1011 (1971) 9
In re Mahoney, 421 F. 2d 742 (1970) 9
In re Musgrave, 431 F. 2d 882 (1970) 9
In re Prater, 415 F. 2d 1393 (1969) 9, 18

Miscellaneous:

ADR Patent U. S. 3,380,029 2
ADR Patent U. S. 3,533,086 2
ADR Patent U. S. 4,066,843 3
Brenner, Commissioner of Patents, "Guidelines to Examination of Programs," 829 Official Gazette of the Patent Office 1, 2 July 6, 1966 19
Burroughs Advertisement, 227 Scientific American 2, July, 1972 7
"Burroughs' Wide Ride With Computer," Business Week, July 1, 1972, pp. 54-58 7
"Chief Programmer Teams," 19 Datamation 58-61, Dec., 1973 10
"Composite Design: The Design of Modular Programs," IBM Poughkeepsie Labs, January, 1973 10
Daly, "Management of Software Development," IEEE Transactions on Software Engineering, IEEE Computer Society, May, 1977, pp. 230-242 13
First National Conference on Software Engineering, Washington, D. C., Sept. 11-12, 1975 10
Galbi, "Proposal for New Legislation to Protect Computer Programming," 17 Bull. Copyright Soc'y. 280, 281 5
Goddard News, Public Information Office of the Goddard Space Flight Center, NASA, Dec. 12, 1966 14
Goetz, "What's Inside the Software Industry," Computer Decisions, July, 1975, p. 18 4
Hewlett-Packard Advertisement, Business Week, March 6, 1978 6

TABLE OF CITATIONS (Continued).

Miscellaneous (Continued): Page

IBM Announcement, Wall Street Journal, Aug. 13, 1972, pp. 43, 49 7
IBM Patent U. S. 3,339,182 2
IEEE International Conference on Software Reliability, April, 1975 10
Jacobs, "Computer Technology (Hardware and Software): Some Legal Implications for Antitrust, Copyrights and Patents," Rutgers Journal of Computers and Law, Vol. 1970, No. 8, p. 50 1
Jacobs, "Patentable Machines—Systems Embodiable in Hardware or Software (The Myth of the Non-Machine)," Proc. Law of Software, 1968, p. B-77, Geo. Wash. Univ. Natl. Law Center 5
"Management Concepts for Top-Down Structured Programming," IBM Technical Report, Feb., 1973 10
"Microcomputers," 232 Scientific American 32-39, May, 1975 .. 19
"Revolution in Programming: An Overview," 19 Datamation 50-52, Dec., 1973 10
Rosen, "Hardware Design Reflecting Software Requirements," 33 Proc. of the Fall Joint Computer Conference, Dec., 1968, Pt. II, pp. 1443, 1444 2
Software Engineering, INFOTECH State of the Art Report 11, 1972 10
"Structured Programming," 19 Datamation 52-54, Dec., 1973 10
"Structured Programming: Top-Down Approach," 19 Datamation 55-57, Dec., 1973 10
"System/360 FLOWCHART," IBM Technical Newsletter, No. N20-0069-0, Sept. 12, 1966 7
The Software Engineering NATO World Conference, Garmisch, Germany, Oct. 7-11, 1968 10
"Top-Down, Bottom-Up, Structured Programming and Program Structuring—A Report on the Apollo Space Shuttle Software," MIT, Charles Draper Labs, Dec., 1972 10

IN THE

Supreme Court of the United States

OCTOBER TERM, 1977

No. 77-642.

LUTRELLE F. PARKER, Acting Commissioner of Patents and Trademarks,

Petitioner,

v.

DALE R. FLOOK,

Respondent.

ON WRIT OF CERTIORARI TO THE UNITED STATES COURT OF CUSTOMS AND PATENT APPEALS

BRIEF AMICUS CURIAE FOR APPLIED DATA RESEARCH, INC. (ADR).

Applied Data Research, Inc. (hereinafter ADR) files this brief *amicus curiae* in opposition to Petitioner, together with letters of consent of the parties in accordance with the Rules of the Court.

QUESTION PRESENTED.

Whether a machine system or process implemented in software [1] is patentable equally with the same machine system or process implemented in hardware.[2]

1. Software, the stored programs used with a general-purpose computer to construct a special-purpose computer.

2. Hardware, the physical equipment or devices forming a general-purpose computer.

For a discussion of relationships of the technologies of hardware and software as they bear on legal issues before the Court, see Jacobs, "Computer Technology (Hardware and Software):

INTEREST OF AMICUS CURIAE.

ADR was incorporated in 1959 and since then has been a small business engaged in the computer field in the marketing of various computer services, proprietary software products and proprietary hardware/software products.[3] ADR presently markets eleven software products and has been awarded two software patents, one of which was the first patent issued for a software product on the market.[4] This first patent was issued to ADR's AUTOFLOW System, which is one of the most successful standardized software products on the market.

ADR also has a line of software/hardware products called STAR. These products monitor telephone lines and are sold as a total machine to large telephone users. ADR developed the software, built display equipment, bought

2. (Cont'd.)
Some Legal Implications for Antitrust, Copyrights and Patents", *Rutgers Journal of Computers and Law*. Vol. 1970, No. 8, p. 50 and "Hardware Design Reflecting Software Requirements," Prof. Saul Rosen, 33 Proc. of the Fall Joint Computer Conference, Dec. 1968, Pt. II, pp. 1443, 1444.

3. ADR has no interest in any patent that may issue on the Flook application.

4. U. S. No. 3,533,086 of Martin A. Goetz, issued October 6, 1970 for "Automatic System for Constructing and Recording Display Charts."

The other ADR patent is U. S. No. 3,380,029 of Martin A. Goetz, issued April 23, 1968 for "Sorting System." The issuance of this patent received considerable publicity as a so-called "first software patent." Actually, this patent was the first one that was *candidly* presented as implemented by software programming. Prior to that, hardware companies had obtained such patents by the stratagem (called Fabian Strategy by the Patent Office Solicitor) of presenting the invention as though implemented by hardware programming, not by software. One example of such a patent is U. S. No. 3,339,182, "Optimum Result Computer" of IBM. The use of this stratagem by the hardware manufacturers and its recognition by the Patent Office and others manifest the common understanding of the engineering equivalence of the software and hardware technologies.

standard mini-computers and with engineers, programmers, draftsmen, and technicians built the STAR systems on a small assembly line.

In January 1978, ADR was granted a patent for its STAR products;[5] software is used in the invention of this patent to build the complete control system of a complex of telephone monitoring operations for the improvement of telephone service that could not otherwise be performed. The software control system of the STAR products is comparable to the use of software controls for the automation of an oil refinery, which is the subject matter of this case.

ADR believes that an adverse decision in this *Flook Case* may undermine ADR's existing software patents.[6]

ADR is a manufacturer of special-purpose computers created by using ADR software with standard general-purpose computers available from hardware manufacturers such as IBM. Due to the nature of the software technology, a small business like ADR can engage in such a manufacturing business and successfully compete in the open market with the giant hardware manufacturers. For example, IBM also markets software products, much of it ostensibly "free," though its principal business is nominally (as a reflection of its hardware pricing strategy) the manufacture of computers based on hardware programming.

The following information reflects ADR's experience in developing and marketing its "software" products: The initial cost to develop a software product ranges between

5. U. S. Patent No. 4,066,843 of David Waks, et al., issued January 3, 1978 for "Telephone Circuit Monitoring System." ADR has recently sold the business of its STAR product line.

6. Software patents are patents on special-purpose machine systems that, because of technological and economic reasons, are constructed by means of stored programs (*i.e.*, software) and a general-purpose computer.

$50,000 and $500,000.[7] The annual cost to service, maintain and enhance each product is about equal to the development costs.[8] It usually takes 1 to 2 years to develop a product to the point where it is marketable. At least 1 year on the market is required before the number of sales for a particular product reaches a satisfactory level in terms of profitability. Software products are expected to remain marketable for at least 5 years, and must be continually improved to stay competitive during that period. Certain software products, however, may be useful for as many as 10 years or more. The obsolescence factor is more dependent upon the life-span of the class of hardware with which a software product operates than on the functioning of the software program itself. In summary, large investments must be made prior to marketing a software product, and continuing investments are required throughout its marketable life-span.[9]

Petitioner and the supporting hardware manufacturers and their association, CBEMA, oppose patents for software implemented inventions on the grounds that software is an "idea," merely a "method of doing business," or a "mathematical algorithm" as though software had a certain ephemeral character, a nonphysical or "non-

7. As an average, about $200,000 per product is usually invested.

8. ADR has invested almost $2 million in the development and maintenance of AUTOFLOW.

9. See Goetz, "What's Inside the Software Industry," *Computer Decisions*, July, 1975, p. 18: "The business of developing and marketing software products is at least as complex as the business of selling computer hardware, and so requires considerable financial and human resources. There are several marketing considerations that clearly reflect the true nature of software products. These include the need for continued investment in a product even after its release, an effective marketing and pricing strategy for each product, and substantial technical support."

machine"[10] character. Software is a machine[11] and not ephemeral nor a "method of doing business," nor an "idea," nor a "mathematical algorithm."

No one case or group of cases can give the Court a picture of the range of complexity of the software technology and the extraordinary variety of machines that have been built and the far larger variety that remain yet to be built. The complexities and intricacies of the software are comparable to those of the general-purpose computer (the hardware) which is an extremely complex machine.

SUMMARY OF ARGUMENT.

The products of competitive hardware and software companies and their opposing economic interests demonstrate that the issues underlying this case are related to the real world of machines and not to "ideas," nor to "mathematical algorithms."

The technological facts reveal that "software patents" are patents for automatic data processing machines and computer machine systems—not patents for "methods of doing business," "mental processes," or "mathematical algorithms." An examination of several examples of innovative software demonstrates the machine nature of the product. Software and hardware are engineering equivalents and any particular program, such as ADR's AUTOFLOW, could alternatively be built in hardware. Hardware machines that "talk" or "read" or "diagnose"

10. See Jacobs, "Patentable Machines—Systems Embodiable in Hardware or Software (The myth of the Non-Machine)," *Proc. Law of Software*, 1968, p.. B-77. Geo. Wash. Univ. Natl. Law Center.

11. As recognized by IBM, ". . . in some respects computer programs and hardware are interchangeable; hence one can say that a new computer program is similar to a new machine." Galbi, "Proposal for New Legislation to Protect Computer Programming," 17 *Bull. Copyright Soc'y.*, 280, 281.

problems in telephone systems are not business methods or mental processes; clearly the same is true of software-built machines which perform these same tasks.

The proposals of IBM and the other hardware manufacturers for copyright or other forms of protection of software would reverse the constitutional policy of providing an incentive for innovation in these "useful Arts."

ARGUMENT.

1. The economics of the computer industry, as well as the opposing economic interests of the competing companies in the computer industry, demonstrate that the real issues involved in this case are machines and machine systems, and not "mathematical algorithms."

The software side, or sub-industry, of the computer industry is itself a multibillion dollar business whose software directs the operation of stored-program general-purpose hardware. Without software programming, the general-purpose hardware is useless [12], but with software, an "extended machine" is produced. Thus, it is the availability of software that has permitted the computer industry to develop. The argument of the hardware manufacturers (CBEMA) about the "free" exchange of programs and "mathematical algorithms" seeks to perpetuate a myth of the "non-machine." These legal arguments stand in strong contrast to their advertised claims for superior machine construction and operation due to the software.[13]

12. To one who sought some criteria for determining the value of such general-purpose hardware without any software, the suggestion was made to call in a junkman, who would ask simply, "What does it weigh?"

13. See Hewlett-Packard advertisement—*Business Week* March 6, 1978, p. 18, "Protecting the Biggest Investment of All—Programming—that's where you really spend money. We know that from

ADR's experience in the computer industry shows the need for software patent protection. Although it is almost impossible for a small software company to compete successfully against IBM's "free" software products, ADR has done so with its AUTOFLOW product for which it has a patent. ADR's software product AUTOFLOW is directed to an automatic system for developing flowcharts of software programming without human intervention. The system that ADR competed with when it placed AUTOFLOW on the market in 1965 was one offered "free" by IBM to its customers. The IBM FLOWCHART [14] system for the IBM/360 hardware was not an automatic system, since the programmer was himself compelled to sketch manually a rough-draft flow diagram of the software programming that he wished to chart. With records of this rough draft of the flowchart, the IBM FLOWCHART software was used to control the hardware to operate in the

13. (Cont'd.)
experience, too, we spent hundreds of man years developing the operating software for the HP 3000."

See Burroughs advertisement in 227 *Scientific American* 2, July 1972, in which Burroughs sells its machines (hardware and software) with the claims of "More powerful and productive computer systems" that offer *inter alia*, "A comprehensive library of program products for major savings in program development time and related costs" and of "A more powerful and productive master control program" that "increases throughput 10 to 20% . . . [and] confirms Burroughs leadership in control program technology." See also the news article, "Burroughs' Wide Ride With Computer," *Business Week*, July 1, 1972, pp. 54-58.

An IBM announcement (in the *Wall Street Journal*, August 3, 1972, pp. 43, 49) of its new machine technology also stresses the machine character of the software in offering its product for sale: "System 370 virtual storage is created by circuitry and advanced programming that link slower but far less costly peripheral disk storage to main storage in the computer." Certainly, it is not an "idea" that, together with circuitry, does the linking of the disk storage to the main storage.

14. See *IBM Technical Newsletter*, "System/360 FLOWCHART," No. N20-0069-0, Sept. 12, 1966.

nature of a drafting machine to draw a finished flowchart. Unlike the IBM "free" software, ADR's AUTOFLOW is fully automatic, the programmer does not have to draw the layout of the flowchart because the machine "plans" it automatically, and thereafter automatically draws the flowchart.

In marketing its AUTOFLOW product, ADR found the potential IBM/360 market totally saturated with IBM's flowcharting program. ADR initially experienced considerable resistance on the part of IBM computer users against paying for a flowcharting program when they got one "free" from IBM, albeit the flowcharting was not automatic. ADR was compelled to invest heavily in educating the market as to the difference between the semi-manual system of IBM and the fully automatic system of ADR, even though IBM's semi-manual flowcharting was not competitive, neither technically nor in the final flowchart that was achieved. ADR was ultimately successful in the marketing of its product, in one of the first and few instances of such successful competition with IBM's "free" software.

Without reasonable prospects of obtaining a patent, ADR would have been confronted with the situation that IBM could use the concepts of the AUTOFLOW system (for which a patent was finally granted) and develop a system similar to AUTOFLOW and give it away "free." Were that to have happened, ADR's AUTOFLOW business would have been destroyed. ADR has sold or leased more than 1900 units of its AUTOFLOW software product, with gross sales of about 15 million dollars.

The hardware manufacturers have never contested ADR's AUTOFLOW patent by infringing it. Certainly, their arguments before this Court about "methods of doing business," "algorithms" and "ideas" would have no credibility in a defense of an infringement suit, for the issues

would be clear that both ADR and the infringer were both engaged in the sale of automatic machines.

The hardware manufacturers such as the members of *amicus* CBEMA supporting Petitioner have obtained patents for their hardware programming developments; these are the products they price in the bundle of hardware and software that they sell. However, the hardware manufacturers do not wish to be restrained in any way by software patents that companies such as ADR may obtain; for such patents would enable small software companies, as in the case of AUTOFLOW, to enter into competition with the hardware manufacturers even as against the "free" software.

Thus, the software business, in every respect, is concerned with the sale and use of automatic machines and not with "methods of doing business" or "ideas" or "mathematical algorithms."

2. The Facts of the Technology and the Findings of Fact of the C. C. P. A. in Its Predecessor Cases Are That a Programmed Digital Computer Is a Special-Purpose Computer or Machine That Performs Machine Functions, Not a "Mathematical Algorithm".

The C. C. P. A. findings of fact in prior cases [15] apply to the present case. They are: 1) that general-purpose hardware is structured by software into a special-purpose computer, and the structure of this special-purpose machine is different from that of the general-purpose machine and is determined by the inserted software; and 2) that hardware and software forms of programming are engineering equivalents. Once the facts are recognized, the

15. *In re Prater*, 415 F. 2d 1393 (1969); *In re Bernhart*, 417 F. 2d 1395 (1969); *In re Mahoney*, 421 F. 2d 742 (1970); *In re Musgrave*, 431 F. 2d 882 (1970); *In re Foster*, 438 F. 2d 1011 (1971).

patentability of the special-purpose machine follows directly from the statutory provision that machines are patentable subject matter. The briefs of Petitioner and CBEMA, the association of the hardware manufacturers, do not deny these findings of fact, but by predicating their attack against the patentability of software programming on "ideas," "laws of nature," or "mathematical algorithms" the hardware manufacturers are indirectly attacking these findings and are, in effect, seeking their reversal by this Court.

The published statements of industry experts of all kinds over the past 10 years [16] demonstrate that software is a formal technology for constructing machine systems. Additional evidence can be found in the formal education process that now takes place in universities. Degrees in computer science are now offered at the bachelor, master and doctorate levels through engineering departments of a large number of universities. Such degrees

16. Software Engineering Conferences, Reports and Articles:
a. First National Conference on Software Engineering—Sept. 11-12, 1975, Washington, D. C.
b. IEEE International Conference on Software Reliability—April 1975.
c. Composite Design; The Design of Modular Programs, IBM, Poughkeepsie Labs, January 1973.
d. Management Concepts for Top-Down Structured Programming. IBM Technical Report—February 1973.
e. Revolution in Programming; Structured Programming; Structured Programming: Top-Down Approach; Chief Programmer Teams—*Datamation* Magazine, Dec. 1973, 50-61.
f. Top-Down, Bottom-up Structured Programming and Program Structuring.—A Report on the Apollo Space Shuttle Software—MIT, Charles Draper Labs, Dec. 1972.
g. Software Engineering. A collection of software engineering articles by industry and university leaders, INFOTECH; State-of-the-Art Series, 1972.
h. The Software Engineering NATO World Conference—October 7-11, 1968 Germany.

cover, for the most part, the computer engineering disciplines and techniques of building software systems.

From ADR's practical experience with software, it is truly a machine system. ADR personnel are highly experienced and highly educated; they build software machines using a formal method of design that produces a machine architecture of components and functions that interact to form a machine system. The design is generally developed by a chief programmer and a small team of highly skilled programmers who literally build the software system. Such systems whether simple or complex have a machine architecture, which in no way resembles a "method of doing business", an "idea" or "mathematical formula," or a "mathematical algorithm."

Examples of Machine Systems Implemented By Software

The following examples of a variety of automated machine systems implemented by software (those of ADR's two software patents and its pending patent application and two others) contradict the assertions that software inventions are "ideas" or "mathematical algorithms."

A. *AUTOFLOW Flowcharting System.* One unique feature of the software technology is the capability of making "decisions," which has led to the mistaken impression that computers "think." This "decision" feature reveals that a special-purpose computer built by a stored computer program is not (except in a trivial sense) merely a sequence of hardware circuits (or "instructions"). In practice, the program structures many interrelated sequences of those circuits. It is not at all like the player piano that invariably plays the same sequence of chords directed by the holes in the piano roll. The "decision" [17]

17. The term "decision" is a rather pretentious one to describe a machine control operation or "selection" which controls processing

capability enables the program to jump from one sequence to another and back again depending on the results of the processing of the data at different stages. Thus the sequences of operations vary with the nature of the data being processed.[18]

The ability to perform "decisions" by means of a computer program controlling a general-purpose computer is an underlying feature that makes possible the building of so many different types of machines by software. This "decision" feature does not exist in any way in the player piano, which has a fixed, invariable mode of operation built into its mechanisms. If the piano roll has any analogy to the computer, it corresponds to the data supplied to a programmed computer, but even this analogy is superficial at best, for the player piano reproduces without modification the exact sequence of notes specified by the sequence of holes in the piano roll. However, the data in a programmed computer is processed and the sequence of processing varies under the control of the "decision" circuitry as directed by the software. Thus, a player piano is nothing but a piano while a general-purpose computer is converted to a special-purpose machine by software, with the structure of that machine being conveyed by the software.

The alternative paths of data processing resemble a maze with an almost infinite number of alternative

17. (Cont'd.)
to follow one of a number of alternative paths of operation. The criteria for selection are the intermediate results of data processing. The technical jargon in the computer field, which is often quite anthropomorphic in suggesting human characteristics where none exists draws upon the only available terminology that we have, and uses existing words with special machine meanings. The hardware manufacturers' arguments about "ideas" are largely and speciously based on this anthropomorphic terminology.

18. See Jacobs, fn. 2, *supra*, for a discussion of an equivalent hardware technique corresponding to the software "decisions" for producing alternative paths of processing.

choices. For example, if there are 50 specific decision points, the number of possible paths is 2^{50} or over a thousand trillion choices (and more than 50 decision points are common within computer programs). The control of data as it passes through the logical maze is not a function of the hardware circuits, but rather a function of the particular software that directs the hardware.

The hardware circuits are so elementary and universal in their functions that it is only in their combinations in intricate sequences determined by the software that the features of the special-purpose computer are uniquely established. This is the same characteristic found in the building of any machine out of a combination of structural elements. As the computer (or television) circuits are built out of combinations of transistors, resistors and capacitors, so a special-purpose computer is built out of combinations of the general-purpose computer's functional circuits.

These multidirectional mazes of alternative paths of processing are charted automatically by the AUTOFLOW system.[19] It cannot "think" and therefore does not know if it is flowcharting a sorter program, a program for analyzing printed characters (*e.g.*, to read and sort mail in a post office), a program for synthesizing voice (so that the blind can receive communications from a computer), or any of

19. The similarity of this maze of "decisions" in a program to hardware logic gates is shown in a paper by E. B. Daly "Management of software development," IEEE Transactions on Software Engineering, IEEE Computer Society, May 1977, pp. 230-242:

> "*Software development is slower.* Hardware logic gates, as measures of complexity, are comparable to branching [decisions] statements in programs, he says. Comparing hardware development projects to software development projects of about the same complexity, he found that it took about twice as many man-hours to develop one instruction as it did to develop one logic gate. Further, it took *four* times as many man-hours to do design maintenance (corrective maintenance) for software as it did for hardware."

a large variety of other programs that a computer center may have (for directing its single set of hardware). The data processing paths of AUTOFLOW are themselves configured in a complex multidirectional maze.

The AUTOFLOW patent,[20] therefore, is directed to a special-purpose computer which in no sense is a "mathematical algorithm." The United States Government, one of the most enthusiastic customers of AUTOFLOW, has encouraged its use because it is not like a manual or mental process: ". . . manually drawn flowcharts often reflect what the programmer intended to have the program do rather than what he coded it to do. By the use of AUTOFLOW, computerized flowcharts are produced which precisely describe what the programmer has coded."[21] Thus, AUTOFLOW is a unique machine system which is constructed with software programming and the general-purpose computer hardware with which it cooperates in a synergistic way to overcome the hardware's limitations and achieve the desired flowcharting in a useful form by practical means. This machine system is clearly not a "mathematical algorithm."[22]

B. *ADR's Sorter Patent.* A second ADR patent[23] is concerned with the most common form of automatic data processing in the commercial field, that of sorting data (*i.e.*, placing data records in numerical or alphabetical sequence). ADR's sorting patent defines a machine in which the sorting is performed on strings of records which

20. See fn. 4, *supra.*

21. Goddard News, Public Information Office of the Goddard Space Flight Center, NASA, Dec. 12, 1966, pp. 3, 7.

22. At ADR, in the development of software products, the special mathematical skills of mathematicians are not used.

23. See fn. 4, *supra.*

are recorded on magnetic tapes which can be "read" or "played back" only when moving in the forward direction.

The machine sorts strings of records on successive tapes then merges the sorted strings of records and stores the merged records on a tape. The next set of records is similarly sorted and merged, and so on until the tapes all have merged strings of records, whereupon they are merged and stored on one tape. This alternation between sorting and merging is repeated until all of the records are sorted.

This sort-merge construction solves a machine problem of achieving efficient operation where the magnetic tapes can be read while moving in one direction only, and the rewinding of tapes occurs during time periods when its records are not needed for sorting or merging. For that reason it is clear that the software achieves a unique machine construction, and is not an "idea" or "method of doing business" or other non-machine.

Sorting data by a "mental process" would never be done in the way ADR's sorting machine does it. The entire purpose of the patented system is to perform operations at high speed, at millionths and thousandths of a second, and to manipulate the tapes in a certain way to overcome the unique limitations of being readable in one direction only. This machine system is clearly neither mathematics, nor a "mathematical algorithm."

C. *The STAR System.* ADR has developed a machine for detecting, reporting and identifying failures and malfunctions that occur in a telephone network of a large number of telephone lines. This system requires, in addition to special circuitry that connects to the telephone lines, a complex control system for automatically monitoring and "interpreting" the progressive changes in state from the initiation of each call on each line to the failure of the call due to a fault or to normal completion or interruption.

This control system is constructed by a stored program and general-purpose computer, and its complexity stems from its ability to simultaneously follow the signals on a large number of telephone lines. The software-controlled machine directs the circuitry to determine which telephone line to monitor at what *millisecond* in time, and provides the type of overall supervision and control that would be associated with an automated factory. The STAR System performs the routine, tedious tasks that might otherwise have to be performed by a large number of people, but which could not be performed as effectively by them, and reduces the amount of circuitry that must be used. The STAR System operates in an entirely different way from that used by humans to monitor operating conditions on telephone lines, ensures complete privacy of those talking on the lines, and because of its great complexity, it could not be practically achieved in the present state of the art with a construction using all hardware programming. It is clearly not a mathematical process, nor a "mathematical algorithm."

D. *Recognition of Printed Characters.* Machine systems have been developed for automatic reading of printed documents. The machines consist of: 1) an input device which produces electrical signals corresponding to the patterns of black and white print that form a printed symbol (*e.g.*, an optical sensor); 2) an analyzer for processing the signals to identify the symbol; and 3) an output device (*e.g.*, a printer or a control device).

The character recognition analyzer may take a wide variety of forms, independent of the types of input and output devices, and it may be embodied either in software or hardware or some combination of software and hardware, depending on various engineering considerations. Generally, hardware is faster, but it suffers from the disadvantages of requiring a larger investment to tool up for mass production, and it may be too expensive for more complex systems.

The software approach for the character recognition analyzer is more flexible and permits a company to build a machine and to test it out before finalizing the hardware design. In addition, the company can place the machine on the market in the software form for commercial use, and be able readily to modify, enhance and debug it as field use develops. As this technology developed and became more complex, the machine analysis, at least initially, was implemented by software.

The hardware and software forms of a character recognition analyzer are engineering equivalents. The machine structure of a particular analyzer would be generally the same whichever form of programming, hardware or software, were used, so from the standpoint of patent protection, it should make no difference which engineering form is chosen. Hardware manufacturers such as IBM have for many years obtained patents for their character recognition analyzers, which they market in hardware form. A small company should be able to obtain a patent for an innovative character recognition analyzer which initially is embodied in software form, and which subsequently, by that small company or by a hardware manufacturer, may be embodied in hardware form.

In no way is a machine system for recognizing letters and numbers a "mental process," or a "mathematical algorithm." Whether the analyzer is hardware- or software-embodied, a patent would certainly not cover the human process of reading, nor any business method, but only a machine structure.

E. *Voice Assembly.* Digital computer techniques have been developed for assembling speech from prerecorded syllables in which parts of words are stored in

a computer memory, and a special-purpose computer is used to assemble complete words and transmit them to a human operator (*e.g.* in a bank, in response to an inquiry about a depositor's bank balance, or a blind man's request for data from a computer data bank). Such a special-purpose computer may be implemented by software combined with a general-purpose computer, or by hardware programming, or by an appropriate combination of the two. The particular form of embodiment, hardware or software, would not affect the machine system or its functions. In neither case would the patent cover mathematics or a "mental process" (that is, a human process for the assembly of speech) or a "mathematical algorithm."

Summary. In each of these examples of machines (an automatic flowcharter, a sorter, a telephone monitoring system, character recognition equipment, voice assembly equipment) the hardware used may be the same (*e.g.*, the general-purpose computer of any of the hardware manufacturers). The hardware is uniquely structured by the software of these widely different machines as required, without rewiring the hardware and without otherwise changing the hardware or its basic operations. In each of these cases, it is clear, not only is there a machine system distinctly different from a human "mental process," but any patents on these systems would not cover mathematics or prevent the human performance of any "mathematical algorithm." This is especially clear in the ADR AUTOFLOW and Sorter patents, where the inventions are defined in the claims as machines, rather than machine processes,[24] and in the STAR patent where both machine and process claims were granted.

24. In accord with the engineering equivalence of software and hardware, is the C. C. P. A. holding in the *Prater* and *Bernhart* cases, *supra*, note 8, that it is appropriate to define the invention

3. AUTOFLOW is a Machine System That Could Be Converted Into a Handwired Hand-held Machine But for Economic Reasons.

In 1965, ADR conceived of the AUTOFLOW "two-dimensional system for constructing and recording display charts" and in 1970 received a patent on this software machine system.[25]

With the advent of microcomputers and integrated circuits, it is possible to build an AUTOFLOW machine system, completely in hardware, based on the principles covered in the patent. Each of the functions, or groups of functions, could be put into one or more "chips" and the entire machine logic would be the size of today's calculator and would form a "black box."[26] Input/output units in the form of small tape reels (as found in hand-held tape recorders) would be appended to this "black box." In today's technology, the entire machine could be put into a 6" x 12" x 2" or smaller surface and held in one's hand. The price of such a unit might be inordinately high, however. The unit price must reflect the fact that integrated circuits are expensive to design and produce. Generally, a minimum volume of 50,000 units would have to be produced before the cost of development could be recovered.[27]

24. (Cont'd.)
as a machine in patent claims using the statutory claim form specified in 35 U. S. C. 112. The 1966 Patent Office Guidelines to the Examination of Computer Programs, 829 O. G. 1, 2, Aug. 2, 1966, also acknowledge the suitability of defining software as a machine device in this fashion. This procedure was followed by ADR in its patents for AUTOFLOW and the Sorting System.

25. Patent No. 3,533,086, fn. 4, *supra*.

26. A "black box" is a term used for a machine (in a casing or box) when the user does not care and is not concerned about the "internal logic" of the machine. The user is only concerned with the function the machine or "box" performs.

27. *Scientific American*, May, 1975, "Microcomputers," 32, 39.
"These specially made integrated circuits can be justified only when they are produced in volumes large enough (50,000 units, say) to amortize their development costs."

There are reasons other than "volume minimums" to reflect on before business decisions would be made for building a hardware version. AUTOFLOW logic changes to reflect new kinds of input and to incorporate improvements in the output, based on the experience of users. Software, although long-lived,[28] is "volatile," (*i.e.*, easily changed). However, had the software been put into hardware, it would have obsoleted the system at a much quicker pace, which is a common experience with automatic machines of the rigidly fixed type such as hardware.

In conclusion, AUTOFLOW as a machine could theoretically be composed of hardware only and not require the use of a general-purpose computer for "the functioning of the machine." Depending on the economics and other trade-offs between hardware and software, much of today's software might well be put in hardware, since it is possible to convert all software-constructed machines into hardware machines for they are both formed of *interacting circuits.* The interaction or "dynamic interrelationship" (to use the Government's term)[29] among the circuits of the general-purpose computer that is produced by the software is the structure of the software-constructed machine. The same structure is produced by the "dynamic interrelationship" between the circuits of the hardware-constructed machine. The difference between the two machines is that the "dynamic interrelationship" is achieved by electrical signals in software and by fixed wire connections in hardware. As the

28. AUTOFLOW, a product of the mid-1960's is still viable. IBM's operating system, which is built as an "extended machine" of the IBM hardware, and much of today's software originated over 10 years ago.

29. "We acknowledge the dynamic interrelationship between general purpose digital computers and the software programs." (Pet. Br. 36 in *Dann v. Johnston*, No. 74-1033).

latter is patentable, so is the equivalent software; for one who builds machines with software is also working in the "useful Arts."

4. The Proposals of IBM and the Other Hardware Manufacturers for Copyright or Other Forms of Protection of Software Would Reverse the Constitutional Policy of an Incentive to Innovation in the "Useful Arts."

IBM and the other hardware manufacturers represented by CBEMA have fully appreciated the technological equivalence of hardware and software and have built their hardware and software on that technological principle.[30] However, they propose that the patent system continue to be used for legal protection of their hardware systems, but not for the equivalent software. Instead of patents, they propose that some registration system (copyright or other forms) be used for the software type of machine.

The hardware manufacturers would turn upside down the constitutional policy underlying the patent system. The grant of patents for a machine is limited to the *innovation* in that machine, as the Court has unequivocally stated (*Graham v. Deere*, 383 U. S. 1, 6):

> The Congress in the exercise of the patent power may not overreach the restraints imposed by the stated constitutional purpose. Nor may it enlarge the patent monopoly without regard to the innovation, advancement or social benefit gained thereby. Moreover, Congress may not authorize the issuance of patents whose effects are to remove existent knowledge from the public domain, or to restrict free access to materials already available. Innovation, advancement, and things which add to the sum of

30. See fn. 11, *supra*.

> useful knowledge are inherent requisites in a patent system which by constitutional command must "promote the Progress of * * * useful Arts." This is the *standard* expressed in the Constitution and it may not be ignored.

A registration system (whether copyright or otherwise) is intended to protect noninnovative features of the machine system embodied in the detailed coded instructions of the software. Such protection is clearly proscribed by the Constitution. Furthermore, that proposal would have the constitutionally innovative features pass into the public domain. Therefore, the constitutional policy of providing an incentive to "innovation," would not be served by the CBEMA proposals; it would be circumvented.

ADR views itself as an innovative company and wishes to compete on the basis of the improvements that it brings to the software arts. It believes the patent system will supply an incentive for making sorely needed advances to this software technology as it has to others, including that of the equivalent hardware.

The hardware manufacturers, which have substantial monopoly positions in the software that they supply with their hardware,[31] seek a registration system for protection of the non-innovative features of their software and would deny to innovative competitors the incentive that is the basic rationale of the patent system.

Besides stimulating innovation, patent protection will enable other small software companies to compete with

31. See Judge Rich's opinion in *Application of Johnston*, 502 F. 2d 765, 774, revd. on other grounds *sub nom. Dann v. Johnston*, 425 U. S. 219, which notes that the "major hardware . . . manufacturers and their representative associations . . . for economic reasons, [do] not want patents granted *on programs* for their machines." (Emphasis in the original).

the hardware giants, as it did *amicus* ADR. It will end the reliance on trade secrets as the only means to protect innovative software, which has had the effect of inhibiting the free flow of information.

The Patent Office and the hardware manufacturers were unable to persuade the Congress in 1968 to legislate exclusion of computer program inventions from patent protection, and Congress found unpersuasive the Report of the President's Commission on Patents on this issue.

CONCLUSION.

The findings of fact and conclusions of law of the court below are soundly based, and its decision should be affirmed.

Respectfully submitted,

CAROL A. COHEN
Attorney for Amicus Curiae,
Applied Data Research, Inc.